Moab and Torque on Cray XT3

Scott Jackson, *Cluster Resources, Inc.*

ABSTRACT: *Moab and Torque provides a powerful resource and workload management solution for the Cray architectures, combining a highly optimized and capability-rich scheduler/policy engine with a widely-adopted and industry-standard open source resource manager. We will use the Cray XT3 solution as an example, demonstrating how using Moab with Torque to manage the XT3 CPA allocation provides crucial capabilities which have been absent from the legacy batch solution.*

KEYWORDS: Moab, Torque, XT3, batch, scheduling, workload management, Cluster Resources

1. Introduction

This paper discusses the advantages in utilizing a Moab/Torque solution for the batch system on the Cray XT3 architecture. The motivation for such a solution is discussed, along with examples of real-world customers who have these needs. The Moab/Torque solution is proposed and salient advantages are described. A CPA[1] allocation model involving the use of Moab and Torque is compared and contrasted with a previous model utilizing Moab with PBS Pro[2]. The design and implementation of the interfaces between Moab and the Cray XT3 resources is described.

Cluster Resources, Inc.

Cluster Resources, Inc. is a leading provider of workload and resource management software and services for cluster, grid and utility-based computing environments. As the developers of the popular Maui Scheduler and the next generation Moab Cluster Suite®, Moab Grid Suite®, and other associated products, Cluster Resources has come to be recognized as a leader in innovation and return on investment. With well over 5,000 clients worldwide, and drawing upon over a decade of industry experience, Cluster Resources delivers the software products and services that enable an organization to understand, control, and fully optimize their compute resources.

Moab

Moab Cluster Suite is a professional cluster management solution that integrates scheduling, managing, monitoring and reporting of cluster workloads. Moab simplifies and unifies management across one or multiple hardware, operating system, storage, network, license and resource manager environments to increase the ROI of cluster investments. Its task-oriented graphical management and flexible policy capabilities provide an intelligent management layer that guarantees service levels, speeds job processing and easily accommodates additional resources.

Torque.

TORQUE[3] is an open source resource manager providing control over batch jobs and distributed

[1] The compute processor allocator manages and allocates compute processing elements to applications.

[2] PBSPro, the commercial release of PBS can be purchased through Altair.

[3] This product includes software developed by NASA Ames Research Center, Lawrence Livermore National Laboratory, and Veridian Information Solutions, Inc. Visit www.OpenPBS.org for OpenPBS software support, products, and information. TORQUE is neither endorsed by nor affiliated with Altair Grid Solutions, Inc.

compute nodes. It is a community effort based on the original PBS project and, with more than 1,200 patches, has incorporated significant advances in the areas of scalability, fault tolerance, and feature extensions contributed by NCSA, OSC, USC , the U.S. Dept of Energy, Sandia, PNNL, U of Buffalo, TeraGrid, and many other leading edge HPC organizations. This version may be freely modified and redistributed subject to the constraints of the included license.

2. The Motivation

When an organization invests in a Cray XT3 system, it represents a serious and carefully considered investment. Cray XT3 systems attract sophisticated scientific High Performance Computing (HPC) customers with complex technical requirements. Cray XT3 acquisitions often involve government and academic sites which have many different projects and workload types that must be effectively combined in the batch system. These are some of the flagship computer resources with national renown.

The following list illustrates examples of customers for whom a Moab/Torque solution was required:

Red Storm from Sandia National Laboratory runs Moab and Torque on a Cray XT3 with 135 racks and 12,960 AMD Opteron CPUs running the Linux/Catamount operating system. Red Storm peaks at 124.42 teraOPS theoretical performance, with 40 terabytes of DDR memory, 340 terabytes of disk storage and 2.5 megawatts of power and cooling.

Jaguar from Oak Ridge National Laboratory runs Moab and Torque on a Cray XT3 with around 18,000 cores – and a roadmap to achieve a Petaflop performance in a single system by 2009.

An Unannounced Leading Government Site also chose Moab and Torque to run on a Cray XT3 with over 18,000 AMD Opteron cores in roughly 100 racks.

3. The Solution

Within installations such as these, there are several tiers of customers to whom the solution must appeal. The batch system will inevitably play a crucial role in the overall satisfaction of the funding managers, site managers, system administrators and users.

The following, though admittedly an oversimplification, are some of the crucial qualities that must be delivered by a batch system for a sophisticated Cray XT3 customer.

High Utilization/ROI = happy investors

Funding managers have commissioned the procurement of the computer to achieve specific results. They want to be able to show high utilization and return on investment for their constituents. They often want to ensure that system cycles are prioritized for specific workload types and groups. Statistics and reports are important to them to provide evidence of delivered performance and utilization.

Enforce Site Objectives = happy managers

Site managers perform a balancing act between the principals, competing department heads, and the users. They often make heavy use of Service Level Enforcement and Guarantees to apply various qualities of service and fairness criteria to different project groups and workload types. They need flexible policies to meet performance objectives. They need to enforce resource sharing between competing political and technical interests. They can show success in these efforts to their principals via graphical charting tools. With expansion and progress always on their minds, capacity planning reports and simulation capabilities are critical tools.

Manageability = happy admins

System administrators have to translate policy into action. The more powerful and flexible the tools at their disposal, the better they are able to perform their jobs. Much is expected from them, so the more that can be automated to eliminate repetitive work the more time they have to devote to system customization. Powerful diagnostics and monitoring tools are indispensable. The ability to evaluate the impact of new policies without impacting production cycles prevents unnecessary risk or policy stagnation. The more the users can do for themselves the better. Although most admins tend to favor the command line, some of the more abstract or complex analysis or customization is made easier via a graphical administrative interface.

Usability = happy users

Most end users don't want to learn much about the batch system. They want simple and standard

job submission and batch environments that don't change from system to system. There will also always be those power users that require a flexible and powerful set of submission options. These can utilize sophisticated job information and control utilities to great advantage. All want reliable cycle delivery and predictable job execution. A web-based job submission portal is also highly desirable due to users logging in from myriads of different environments.

3.1 Why Moab?

There are many compelling reasons that sophisticated sites are choosing Moab Workload Manager.

System utilization is improved to run between 90-99 percent due to intelligent resource allocation and workload ordering.

Advance Reservations (administrative, standing, job and personal) allow for high utilization around maintenance periods, coexistence of different workload types, enforcement of policy agreements and vastly stretch the capabilities of legacy queues.

Moab enforces ***Service Level Guarantees*** through the use of features such as Quality of Service, flexible priority mechanisms, fairshare and usage throttling.

Through the use of ***Resource Manager Translation***, which lets Moab emulate scripts and job language translation of other resources managers, such as PBS Pro, users can continue to use the batch submission interfaces they are used to

Additionally, Moab can create ***a Grid across your Clusters*** – bringing together different resource manager types, operating systems and architectures. Unifying an organization's clusters in this way helps to improve overall utilization, turnaround, access to a greater variety of resource types, co-allocation of disparate resources and unified batch management.

3.2 Why Torque?

There are also compelling reasons to choose to use Torque with Moab on the XT3.

Torque is an ***Industry Standard Batch System*** that is well understood and familiar to users worldwide.

It is ***free, open source*** and ***commercially supported***.

Torque provides built-in ***underlying support for Moab's advanced features.***

And finally, as we shall explore further in this paper, ***Torque permits Moab to handle the Cray CPA partition creation*** which permits:

- ✓ Better Failure Recovery
- ✓ Reservations (Admin, Standing, etc.)
- ✓ Heterogeneous Resources
- ✓ Node Features

3.3 Moab and Torque

Moab and Torque provide a world class solution to the question of batch environment selection for the XT3. Moab and Torque were chosen by the leading XT3 sites because of their superior advantages in meeting their sophisticated needs.[4]

4. CPA Allocation Model Comparison

When the Cray XT3 was originally introduced, Cray initially offered only a batch solution with Altair's PBS Pro. Moab was later provided to bring sites the advanced scheduling and workload management capabilities they needed.

Some initial motivations for utilizing Moab on top of PBS Pro included:[5]

- ✓ More flexible resource management capabilities
- ✓ A wide variety of highly configurable policies
- ✓ Context-sensitive dynamic policy actions
- ✓ Highly tunable job prioritization
- ✓ Non-disruptive evaluation of new policies
- ✓ Ability to harness resource information from multiple sources
- ✓ Improved Diagnostics

[4] Footnote: Although this paper focuses on the XT3 solution, Moab solutions exist for other Cray architectures such as X1E and XD1 with an XT4 solution on the horizon.

[5] Jackson, D., Maxwell, D., & Jackson, S. (CUG 2006). *Moab Workload Manager on Cray XT3*. For complete text contact info@clusterresources.com.

- ✓ Dynamic failure responses
- ✓ Powerful statistical reporting

It was soon found that although this combination represented a significant improvement to the base case, some serious limitations to Moab's full capabilities under this model became evident.

A plan to address these limitations was soon underway which utilized Torque instead of PBS Pro and enhanced Moab to manage the CPA partition creation and management.

We shall now compare what we will refer to as the *Prior Model* (Moab with PBS Pro) with the *Improved Model* (Moab with Torque).

The Prior Model

In the Prior Model, the PBS Pro pbs_mom handled CPA partition creation and management. Jobs could be submitted to either Moab (via msub) or PBS Pro (via qsub). The Moab Workload Manager would initiate the execution of a job via a qrun command to PBS Pro. PBS Pro dispatched the job to the pbs_mom on the selected yod or login node which created the CPA partition and then executed the job.

Unfortunately, *there was no control over where a particular job was launched*. PBS Pro would simply ask to be allocated size=n nodes and CPA would allocate any available n nodes to the job. There was no way to launch a job on a particular hostlist. This caused a number of limitations to Moab's full capabilities and scheduling optimizations. It was very difficult to properly enforce advance reservations when system and job reservations would be completely ignored after a job was launched. Moab compensated, where possible, by constantly remapping job hostlists and dynamically reshaping the admin and standing reservations, but some reservation dynamics were not optimal.

There was also no support for heterogeneous resources or node-based policies of any type since you could not select nodes for job placement. Additionally, node allocation policies such as running jobs with the closest matching available resources, or running jobs on nodes coming available just-in-time could not be enforced so some of Moab's scheduling optimizations were diminished.

A final drawback in this model was a poor job failure feedback mechanism. When Moab told PBS Pro to start a job, Moab would frequently get a successful return code from the launch only to find out the next time Moab polled that the job was not running. qrun would return success, but if the CPA allocation subsequently failed (such as for a Lustre recovery issue), the job would quietly revert to the idle state. The only way to detect the cause of the failure would be to dig through the pbs_mom or cpa logs.

The Improved Model

In the Improved Model, Moab took charge of creating and managing the CPA partitions. After a job was submitted and was ready to start, Moab created a CPA partition for the job, after which it initiated the execution of a job via the qrun command to Torque.

Since Moab manages the creation of the CPA partition before launching the job to Torque, it can respond immediately and intelligently to any CPA allocation failure conditions. In this way, CPA allocation failures are detected by Moab before the job starts, enabling Moab to intelligently handle the rerouting of jobs to other available nodes. The failure causes can be discerned by Moab, reported to administrators and automatically responded to.

The CPA API supports the ability to allocate a partition across selected processors. With Moab in charge of creating the CPA partitions, it is able to launch jobs on a list of nodes of its choice. This is significant because Moab's scheduling optimizations can now be honored. Node availability policies such as MinResource and LastAvailable can be used to improve utilization and resource matching. Nodes can be selected based on load or the amount of free memory and disk.

In the Improved Model node sharing becomes possible. Moab can allow multiple jobs to run on a single node and these jobs can be subject to constraints such as only allowing jobs from the same user, etc.

The full power of Moab reservations can now be employed in a straightforward manner. Jobs can be placed where desired and will no longer trample existing reservations. Administrative reservations can cordon off nodes for maintenance or special access. Standing reservations can carve off resources for certain workload types or enforcing service level guarantees. Job reservations can help enforce prioritization schemes, fairness policies and performance goals.

The ability to schedule on specific nodes leads to support for heterogeneous resources and node policies. Nodes configured with different disk, memory, swap, architecture, operating system and

other properties can be scheduled in a much more optimal fashion. Node features can be used to assign different labels to nodes to be requested in jobs. Generic resources can be used to assign a consumable resource of arbitrary capacity to various nodes. One can steer jobs to specific nodes or sets of nodes, such as via node lists or node sets. Per node limits, different policies, rules and constraints can be enforced. Moab can route jobs around failing nodes and blocked resources.

Since Moab creates the CPA allocation before the job starts and is able to start jobs on a selected list of nodes, the Improved Model offers great advantages over the Prior Model.

5. Implementation Details

Moab can interact with external resources either by making calls to compiled-in library functions (such as PBS, Torque and CPA libraries), by utilizing its resource manager native interface to invoke scripts that translate between Moab and external commands, or a hybrid of the two.

The interface between Moab and the Cray XT3 batch environment utilizes a hybrid model of both API calls and resource manager native interface scripts found in Moab's tools directory. In this section, we will discuss the interface mechanisms used in the Improved Model. In some cases, function calls are made to Torque and CPA libraries. Some actions are passed through to Torque directly (e.g. Queue Query, Job Submit, Job Cancel). Other actions (e.g. Node Query, Job Query, Job Start) use native interface scripts to aggregate and translate information from external sources such as Torque (e.g. qstat, pbsnodes), compiled CPA commands and the XTAdmin Database.

We will finish by briefly describing the mechanics of six specific actions normally performed by Moab in a common scheduling cycle. The actions we will examine are Node Query, Job Query, Class Query, Job Submit, Job Start and Job Cancel.

Node Query

Moab obtains node information on the Cray XT3 by invoking a script (node.query.xt3.pl) and reading the node information in a name=value format on standard output. The script performs the following steps:

1. Obtain node class information from Torque (qstat –q)
2. Obtain processor information from the XTAdmin database (processor and luster tables)
3. Obtain login and yod node information from Torque (pbsnodes –a)
4. Obtain cpa allocation information from the CPA API (cpa_lookup_nodes)
5. Return node information to Moab on stdout

Job Query

Moab obtains job information by invoking a script (job.query.xt3.pl) and reading the job information in a name=value format on standard output. The script performs the following steps:

1. Obtain job information from Torque (qstat –a)
2. Obtain job tasklist information from the XTAdmin database (partition and partition_allocation tables)
3. Return job information to Moab on stdout

Class Query

The class information can be obtained via a direct call to the pbs_statqueue function via the Torque API.

Job Submit

Jobs are submitted to Torque by invoking the Torque qsub command. The return code and output are captured and parsed by Moab.

Job Start

When Moab wants to start a job, it first attempts to create a CPA allocation via the cpa_create_partition function via the CPA API. If this is successful, the job is started by calling the Torque qrun command. The job status information (success or failure) is returned to Moab.

Job Cancel

Jobs are cancelled via a direct call to the pbs_deljob function via the Torque API.

6. Conclusion

The advanced scheduling and workload management capabilities of Moab and Torque help Cray sites improve their utilization, enforce service level guarantees, improve manageability and usability of the system – features crucially needed by

the type of sites that use the Cray architecture. Taking the Cray XT3 as an example, a model that uses Moab and Torque to manage CPA allocations allows superior support for optimized scheduling, advance reservations, heterogeneous resources, node sharing, and node failure handling.

About the Author

Scott M. Jackson is the Vice President of Software Engineering at Cluster Resources, Inc. He previously worked at IBM as well as MHPCC (DOD) & PNNL (DOE) computing facilities. Jackson is the architect/developer of QBank & Gold (resource allocation management software tools) & has actively participated in the Global Grid Forum developing grid standards and acting as a chair for the Usage Record working Group.

Appendix A:

The following figure displays a number of key integration points between Cluster Resources' Moab Workload Manager product and external batch services.